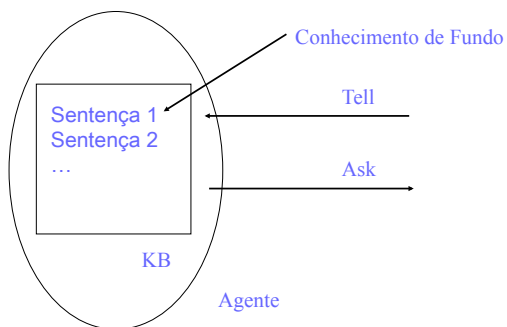
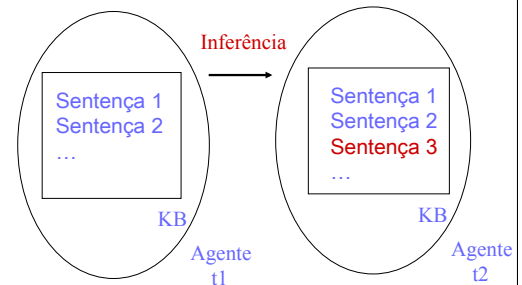
PCS 2428 / PCS 2059
Inteligência Artificial

Prof. Dr. Jaime Simão Sichman
Prof. Dra. Anna Helena Reali Costa

Lógica Proposicional

Agentes Baseados em Conhecimento

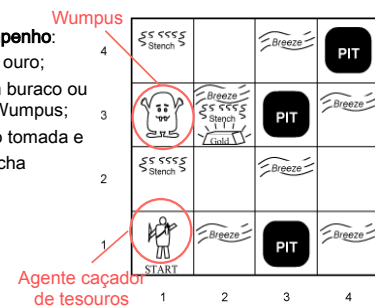
- Como representar conhecimento e como utilizar este conhecimento através de um processo de raciocínio que o torne útil é uma questão central em IA
- Utilização é importante em ambientes que sejam parcialmente observáveis, pois o agente pode combinar o resultado de percepções correntes com um conhecimento anterior (ex: diagnose médica, processamento de linguagem natural)
- Um agente baseado em conhecimento é flexível, pois este pode ser alterado/adicionado

Agentes Baseados em Conhecimento

Agentes Baseados em Conhecimento

Agente Baseado em Conhecimento

função Agente-Baseado-Conhecimento(*percepção*)
retorna uma *ação*
estático: base de conhecimento BC, contador t =0
Tell(*BC*, Sentenças-Percepções(*percepção*,*t*))
ação ← Ask(*BC*, Pergunte-Ação(*t*))
Tell(*BC*, Sentença-Ação(*ação*,*t*))
t ← *t* + 1
retorna *ação*

– Abordagem declarativa

Mundo do Wumpus

Medida de desempenho:
+1000 se pegar o ouro;
-1000 se cair num buraco ou virar refeição do Wumpus;
-1 para cada ação tomada e
–10 por usar a flecha

Descrição do Mundo do Wumpus

- Ambiente: **agente, Wumpus, cavernas (células), buracos, ouro**
- Estado inicial:
  - agente na caverna (1,1) com apenas uma flecha, olhando para a direita
  - Wumpus e buracos em cavernas quaisquer
- Objetivos:
  - pegar a barra de ouro e voltar à caverna (1,1) com vida, para sair
- Percepções:
  - fedor, brisa, luz, choque (contra a parede externa do ambiente) e grito do Wumpus (quando morre) – *vetor: [f, b, l, c, g]*
- Ações:
  - *avançar* para próxima caverna
  - *girar* 90 graus à *direita* ou à *esquerda*
  - *pegar* um objeto na mesma caverna que o agente se encontra
  - *atirar* na direção para onde o agente está olhando (a flecha para quando encontra uma parede ou mata o Wumpus)
  - *sair* da caverna

Caracterização do Ambiente do Wumpus

- Observável:
  - Não, percepção apenas local
- Determinístico:
  - Sim, saídas totalmente especificadas
- Episódico:
  - Não, sequencial ao nível das ações
- Estático:
  - Sim, Wumpus e buracos não se movem
- Discreto:
  - Sim
- Monoagente:
  - Sim, o único agente é o caçador

Codificação do Mundo do Wumpus

	1	2	3	4
4	fedor		brisa	**B**
3	**W**	**O** fedor brisa, luz	**B**	brisa
2	fedor		brisa	
1	**A** início	brisa	**B**	brisa

A – Agente (em (1,1))
W - Wumpus
B - Buraco
O - Ouro

#W, #O = 1
#B = 3
Posições aleatórias (exceto (1,1))

Raciocinando e Agindo no Mundo do Wumpus

Raciocinando e Agindo no Mundo do Wumpus

Representando Conhecimento por Lógica

Lógica Matemática
Formalização de uma linguagem lógica
Sintaxe
Semântica
Inferência
Especificação das expressões legais da linguagem
Significado das expressões da linguagem
Regras que manipulam expressões da linguagem e que preservam alguns aspectos semânticos

Lógica Matemática: Sintaxe
Formalização de uma linguagem lógica
Ex: linguagem aritmética
Sintaxe
α: x + y = 4 é uma fórmula bem formada
β: x 4 y + = não é uma fórmula bem formada
Especificação das expressões legais da linguagem

Lógica Matemática: Semântica
Formalização de uma linguagem lógica
Interpretação 1
Interpretação 2
x = 2
y = 2
Semântica
x = 0
y = 1
Significado das expressões da linguagem
V
F
1 é um modelo para α
α: x + y = 4

Lógica Matemática : Conseqüência Lógica
Formalização de uma linguagem lógica
Interpretação 1
α ⊨ β
Em toda interpretação onde α é V, β é V
Quando α for V, β também precisa ser V
x = 2
y = 2
Semântica
Significado das expressões da linguagem
V
V
1 é modelo para α
α: x + y = 4 ⊨ β: 4 = x + y

Consequência Lógica no Mundo do Wumpus
Modelos de KB
Modelos de α1
KB ⊨ α1
Agente depois do passo 1, em [2,1]
α1: não há buraco em [1,2]
Modelos para presença de buracos em [1,2], [2,2] ou [3,1], dado que não existe nada em [1,1] e uma brisa em [2,1]

Consequência Lógica no Mundo do Wumpus
Modelos de KB
Modelos de α2
KB ⊭ α2
Agente depois do passo 1, em [2,1]
α2: não há buraco em [2,2]
Modelos para presença de buracos em [1,2], [2,2] ou [3,1], dado que não existe nada em [1,1] e uma brisa em [2,1]

Lógica Matemática: Histórico

A lógica tem mais de 23 séculos de história!

- Grécia antiga: Platão, Aristóteles
- George Boole (1815-1864)
  - lógica formal, versão inicial do cálculo proposicional
- Gottlob Frege (1848-1925)
  - primeira versão do cálculo de predicados
- Kurt Gödel (1906-1978)/Jacques Herbrand (1930)
  - procedimento completo para o cálculo de predicados
- Alonzo Church / Alan Turing (1936)
  - cálculo de predicados é indecidível

Lógica Matemática: Histórico

- John McCarthy (1958)
  - lógica de predicados como ferramenta para IA
- Robinson (1965)
  - método da resolução
- Smullyan (1968)
  - método de tableaux
- Colmerauer (1973)
  - Linguagem Prolog

Sintaxe: Proposições

Trata-se do cálculo mais simples que existe.

Uma **proposição** é um enunciado declarativo.

- p: Todo imigrante italiano é palmeirense
- q: Qualquer palmeirense gosta de comer feijoada

Proposições podem ser **verdadeiras** ou **falsas.**

Sintaxe: Conectivos Lógicos

O cálculo de predicados constrói sentenças complexas a partir de proposições simples e de **conectivos lógicos**:

- ¬ : negação (não)
- ∧ : conjunção (e)
- ∨ : disjunção (ou)
- → : implicação (condicional)
- ↔ : bicondicional

Sintaxe do Cálculo Proposicional em BNF

Símbolo inicial: <sentença>
<sentença>::= <sentença atômica> |
<sentença complexa>
<sentença atômica>::= V | F | <símbolo>
<símbolo>::= p | q | r | s | ...
<sentença complexa>::=
¬ <sentença> | (<sentença>) |
<sentença> ∧ <sentença> |
<sentença> ∨ <sentença> |
<sentença> → <sentença> |
<sentença> ↔ <sentença>

Semântica: Princípios Fundamentais

Fórmulas bem formadas s têm um valor verdade (V ou F)

1. **Princípio da Identidade**: Todo objeto é idêntico a si mesmo.
2. **Princípio da não contradição**: Uma proposição não pode ser verdadeira e falsa ao mesmo tempo.
3. **Princípio do terceiro excluído**: Uma proposição é falsa ou verdadeira, não havendo um terceiro caso.

Semântica: Interpretações e Modelos

Interpretação 1
p = V
q = F
F
p ∧ q

O valor verdade de uma fbf é calculado em uma determinada interpretação

Assim, uma fbf pode ter o valor verdade V em uma interpretação e o valor verdade F em outra interpretação

Uma interpretação que torna uma fbf verdadeira é chamada de **modelo** desta fbf

Interpretação 1
p = V
q = V
V
p ∧ q

Semântica: Conectivos Lógicos

Dadas 2 fbfs α e β e uma interpretação I:

1. ¬ α tem valor verdade V em I se e somente se α tiver valor verdade F.
2. α ∧ β tem valor verdade V em I se e somente se α e β tiverem valor verdade V.
3. α ∨ β tem valor verdade F em I se e somente se α e β tiverem valor verdade F.

cont.

Semântica: Conectivos Lógicos

4. α → β tem valor verdade V em I se e somente se α tiver valor verdade F ou β tiver valor verdade V.
5. α ↔ β tem valor verdade V em I se e somente se α e β tiverem o mesmo valor verdade.

Semântica: Tabela da Verdade

Caso uma fbf contenha n sentenças atômicas, existem 2^n interpretações distintas, que podem ser colocadas numa **tabela da verdade.**

	Conectivo **NÃO** Negação		
	p	q	¬p
Interpretação 1 →	F	F	V
Interpretação 2 →	F	V	V
Interpretação 3 →	V	F	F
Interpretação 4 →	V	V	F

Semântica: Tabela da Verdade

Conectivo **OU** Disjunção			Conectivo **E** Conjunção		
p	q	p ∨ q	p	q	p ∧ q
F	F	F	F	F	F
F	V	V	F	V	F
V	F	V	V	F	F
V	V	V	V	V	V

Semântica: Tabela da Verdade

Conectivo Condicional			Conectivo Bicondicional		
p	q	p → q	p	q	p ↔ q
F	F	V	F	F	V
F	V	V	F	V	F
V	F	F	V	F	F
V	V	V	V	V	V

Semântica: Fórmulas Bem Formadas

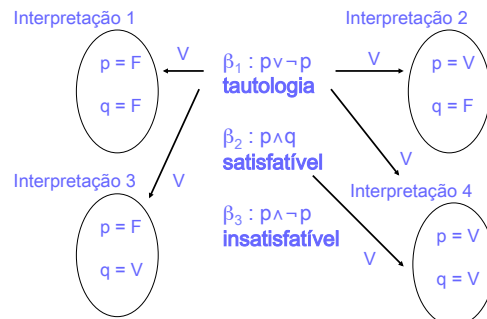
A **semântica** de uma fbf α é definida pela avaliação do seu valor valor verdade em todas as possíveis interpretações, através da atribuição de valores verdade aos átomos que compõem α, levando-se em conta a semântica dos conectivos.

	p	q	p∨q	p∧q	(p∨q)→(p∧q)
Interpretação 1	F	F	F	F	V
Interpretação 2	F	V	V	F	F
Interpretação 3	V	F	V	F	F
Interpretação 4	V	V	V	V	V

Semântica: Fórmulas Bem Formadas

Uma fórmula bem formada α é :

- **Válida (tautologia)** se tiver valor **V** em todas as interpretações;
- **Satisfatível** se tiver valor **V** em alguma interpretação;
- **Contingente** se não for válida nem insatisfatível;
- **Inválida** se tiver valor **F** em alguma interpretação;
- **Insatisfatível (contradição)** se tiver valor **F** em todas as interpretações.

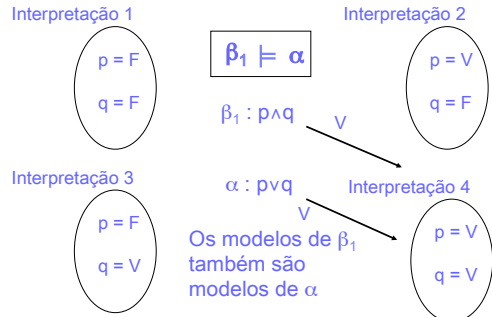
Semântica: Fórmulas Bem Formadas

Consequência Lógica

Dadas as fbfs $\beta_1, \beta_2, \ldots, \beta_n$ e α, diz-se que α é **consequência lógica** de $\beta_1, \beta_2, \ldots, \beta_n$ se e somente se para quaisquer interpretações onde $\beta_1, \beta_2, \ldots, \beta_n$ forem simultaneamente verdadeiras, α também é verdadeira

$$\beta_1, \beta_2, \ldots, \beta_n \models \alpha$$

Entretanto, este procedimento pode ser custoso!

Felizmente, existem propriedades sintéticas que podem auxiliar a verificação ou geração de uma consequência lógica.

Consequência Lógica

Consequência Lógica

Teorema da Dedução

Dadas as fbfs $\beta_1, \beta_2, \ldots, \beta_n$ e α, α é consequência lógica de $\beta_1, \beta_2, \ldots, \beta_n$ se e somente se a fbf

$$\beta_1 \wedge \beta_2 \wedge \ldots \wedge \beta_n \rightarrow \alpha$$

for uma tautologia.

Teorema da Contradição

Dadas as fbfs $\beta_1, \beta_2, \ldots, \beta_n$ e α, α é consequência lógica de $\beta_1, \beta_2, \ldots, \beta_n$ se e somente se a fbf

$$\beta_1 \wedge \beta_2 \wedge \ldots \wedge \beta_n \wedge \neg \alpha$$

for uma contradição.

Consequência Lógica

Provar que $p \wedge q \models p \vee q$

Tautologia

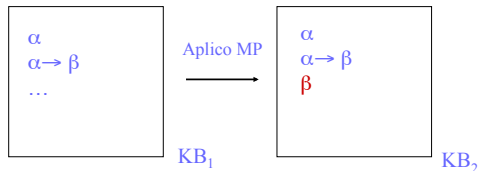
	p	q	p∧q	p∨q	p∧q → p∨q
Interpretação 1	F	F	F	F	V
Interpretação 2	F	V	F	V	V
Interpretação 3	V	F	F	V	V
Interpretação 4	V	V	V	V	V

O ideal seria poder "descobrir" uma tautologia sem precisar analisar 2^n linhas de uma tabela da verdade!

Regras de Inferência

Exemplo: Modus Ponens (MP)

$$\alpha, \alpha \rightarrow \beta \models \beta$$

KB_1		KB_2
α $\alpha \rightarrow \beta$ …	Aplico MP →	α $\alpha \rightarrow \beta$ β

Adiciono ao meu conjunto de sentenças suas consequências lógicas!

Regras de Inferência

Modus Ponens	$\alpha, \alpha \rightarrow \beta \models \beta$
Modus Tollens	$\alpha \rightarrow \beta, \neg\beta \models \neg\alpha$
Silogismo Hipotético	$\alpha \rightarrow \beta, \beta \rightarrow \gamma \models \alpha \rightarrow \gamma$
Silogismo Disjuntivo	$\alpha \vee \beta, \neg \alpha \models \beta$
Simplificação	$\alpha \wedge \beta \models \alpha$
Conjunção	$\alpha, \beta \models \alpha \wedge \beta$
Adição	$\alpha \models \alpha \vee \beta$
Contraposição	$\alpha \rightarrow \beta \models \neg\beta \rightarrow \neg\alpha$
Resolução	$\alpha \vee \beta, \neg \alpha \vee \gamma \models \beta \vee \gamma$

No caso do Modus Ponens, lê-se: Se α for verdade e também $\alpha \rightarrow \beta$ for verdade, então obrigatoriamente β deve ser verdade.

Demonstrações por Dedução

Exemplo: Verificar se o argumento lógico é válido:

- Se as uvas caem, então a raposa as come
- Se a raposa as come, então estão maduras
- As uvas estão verdes ou caem

Logo

- A raposa come as uvas se e só se as uvas caem

Demonstrações por Dedução

Nomeando:
p: as uvas caem
q: a raposa come as uvas
r: as uvas estão maduras

C1: $p \rightarrow q$	1a. premissa
C2: $q \rightarrow r$	2a. premissa
C3: $\neg r \vee p$	3a. premissa
C4: $r \rightarrow p$	(Subst. Equiv. C3)
C5: $q \rightarrow p$	(Silog. Hip. C2 e C4)
C6: $p \rightarrow q \wedge q \rightarrow p$	(Conjunção C1 e C5)
C7: $p \leftrightarrow q$	(Subst. Equiv. C6)

Demonstrações por Dedução

Prova Direta: Parte-se das premissas e chega-se à conclusão, normalmente utilizando Modus Ponens. Usa-se o Teorema da Dedução. É chamada também de **dedução lógica**.

Prova por Contradição: Utilizando-se o Teorema da Contradição, nega-se a conclusão e prova-se que a conjunção das premissas com esta conclusão negada é insatisfatível. É chamada também de **refutação**.

Um caso especial é a chamada **prova por contra-posição:** prova-se que a negação da conclusão implica logicamente a negação das premissas

Demonstrações utilizando Resolução

A regra de inferência da resolução é muito utilizada na prática, pois além de ser uma regra de inferência **correta** ela também é **completa** para a demonstração por refutação.

Para poder aplicar a regra da resolução, o conjunto de fbfs deve ser transformado numa conjunção de **cláusulas.**

Uma **cláusula** é uma disjunção de literais (átomos ou átomos negados).

Demonstrações utilizando Resolução

Prova-se que qualquer fbf do cálculo proposicional pode ser transformada num conjunto de cláusulas equivalente, através dos seguintes passos:

1. Substituir $\alpha \leftrightarrow \beta$ por $(\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$
2. Substituir $\alpha \rightarrow \beta$ por $\neg\alpha \vee \beta$
3. Colar as negações nos átomos, utilizando as equivalências $\neg(\neg\alpha) \equiv \alpha$, $\neg(\alpha \wedge \beta) \equiv \neg\alpha \vee \neg\beta$ e $\neg(\alpha \vee \beta) \equiv \neg\alpha \wedge \neg\beta$
4. Distribuir as disjunções pelas conjunções, utilizando $\alpha \vee (\beta \wedge \gamma) \equiv (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$

Demonstrações utilizando Resolução

Transformar para a forma clausal a fbf:

$$\neg((p \vee r) \rightarrow s) \wedge (\neg p \vee ((p \wedge r) \leftrightarrow s))$$

1. Substituir o conectivo bicondicional
$\neg((p \vee r) \rightarrow s) \wedge (\neg p \vee (((p \wedge r) \rightarrow s) \wedge (s \rightarrow (p \wedge r))))$

2. Substituir o conectivo condicional
$\neg(\neg(p \vee r) \vee s) \wedge (\neg p \vee ((\neg(p \wedge r) \vee s) \wedge (\neg s \vee (p \wedge r))))$

3. Colar as negações nos átomos
$\neg((\neg p \wedge \neg r) \vee s) \wedge (\neg p \vee (((\neg p \vee \neg r) \vee s) \wedge (\neg s \vee (p \wedge r))))$
$(\neg(\neg p \wedge \neg r) \wedge \neg s) \wedge (\neg p \vee (((\neg p \vee \neg r) \vee s) \wedge (\neg s \vee (p \wedge r))))$
$((p \vee r) \wedge \neg s) \wedge (\neg p \vee (((\neg p \vee \neg r) \vee s) \wedge (\neg s \vee (p \wedge r))))$

Demonstrações utilizando Resolução

4. Distribuir as disjunções pelas conjunções

((p∨r)∧¬s)∧(¬p∨ (((¬p∨¬r)∨s)∧((¬s∨p)∧(¬s∨ r))))
((p∨r)∧¬s)∧(¬p∨ ((¬p∨¬r∨s)∧(¬s∨p)∧(¬s∨ r)))
((p∨r)∧¬s)∧((¬p∨¬p∨¬r∨s)∧(¬p∨¬s∨p)∧(¬p∨¬s∨ r))
(p∨r)∧¬s∧(¬p∨¬r∨s)∧~~(¬p∨¬s∨p)~~∧(¬p∨¬s∨ r)
(p∨r)∧¬s∧(¬p∨¬r∨s)∧~~(¬p∨¬s∨ r)~~
(p∨r)∧¬s∧(¬p∨¬r∨s)

Portanto, ¬((p∨r)→s)∧(¬p∨((p∧r)↔s)) é equivalente ao conjunto de cláusulas

{(p∨r), ¬s, (¬p∨¬r∨s)}

Demonstrações utilizando Resolução

Uma demonstração **por resolução** consiste em se aplicar repetidamente a regra da resolução a pares de cláusulas, gerando-se novas cláusulas até que se chegue à conclusão.

A nova cláusula gerada é chamada de **resolvente** das cláusulas originais.

Exemplo:

p1:	a V b	(premissa)
p2:	¬a V c	(premissa)
p3:	¬c V d	(premissa)
C4:	b V c	(Resolução p1 e p2)
C5:	b V d	(Resolução p3 e C4)

Demonstrações utilizando Resolução

Um resultado interessante é que quando a regra é aplicada a duas cláusulas quaisquer, sendo uma delas unitária (apenas um literal), o resolvente terá sempre um literal a menos. Tal fato é explorado nas **refutações por resolução,** e a contradição é detectada pela dedução da **cláusula vazia ({}).**

Exemplo:

Provar, por uma refutação por resolução, o seguinte argumento lógico:

p, p → q, q → r ⊢ r

Demonstrações utilizando Resolução

Prova Direta:

p1:	p	(premissa)
p2:	¬p ∨ q	(f. clausal premissa)
p3:	¬q ∨ r	(f. clausal premissa)
C4:	q	(Resolução p1 e p2)
C5:	r	(Resolução p3 e C4)

Refutação:

p1:	p	(premissa)
p2:	¬p ∨ q	(f. clausal premissa)
p3:	¬q ∨ r	(f. clausal premissa)
p4:	¬r	(neg. da conclusão)
C4:	q	(Resolução p1 e p2)
C5:	r	(Resolução p3 e C4)
C6:	{}	(Resolução p4 e C5)

Cláusulas de Horn

Disjunções com no máximo um literal positivo:

¬b1,2 ∨ ¬b2,1 ∨ B1,1

~~¬b1,1 ∨ b1,2 ∨ B1,1~~

Podem ser reescritas utilizando a implicação:

b1,2 ∧ b2,1 → B1-1

corpo — cabeça

Posso ainda representar fatos e restrições de integridade:

¬b1,2 ∨ ¬b2,1

W1-1 ∧ W1-2 → F

Inferências com Cláusulas de Horn

- Existem algoritmos especializados para realizar inferências em bases de conhecimento compostas por cláusulas de Horn
- São denominados de encadeamento para frente (forward chaining) e encadeamento para trás (backward chaining)
- Decidir se uma cláusula é consequência lógica de uma base utilizando cláusulas de Horn pode ter uma implementação <u>linear</u> em função da BC

Base de Conhecimento Proposicional para o Mundo do Wumpus

A *Base de Conhecimento* consiste em:

- sentenças representando as percepções do agente
  - Existe brisa na caverna (1,2): b1,2
  - Existe fedor na caverna (1,3): f1,3
- sentenças válidas implicadas a partir das sentenças das percepções (conhecimento do domínio)
  - Se existe brisa na caverna (x,y), então existe um buraco em alguma caverna adjacente
  - $bx,y \Leftrightarrow Bx,y+1 \lor Bx,y11 \lor Bx+1,y \lor Bx\text{-}1,y$
  - Se existe fedor na caverna (x,y), então o Wumpus se encontra em alguma caverna adjacente
  - $fx,y \Leftrightarrow Wx,y+1 \lor Wx,y11 \lor Wx+1,y \lor Wx\text{-}1,y$
  - $W1,1 \lor W1,2 \lor ... \lor W4,4$
  - $\neg W1,1 \lor \neg W1,2$
  - Devemos ter 120 sentenças deste último tipo!

Base de Conhecimento Proposicional para o Mundo do Wumpus

- Com base nas percepções do estado abaixo, a BC deverá conter as seguintes sentenças:

$\neg f1\text{-}1 \quad \neg b1\text{-}1$
$\neg f2\text{-}1 \quad b2\text{-}1$
$f1\text{-}2 \quad \neg b1\text{-}2$

V - caverna visitada

Base de Conhecimento Proposicional para o Mundo do Wumpus

- O agente também tem algum conhecimento prévio sobre o ambiente, *e.g.:*
  - se uma caverna não tem *fedor*, então o Wumpus não está nessa caverna, nem está em nenhuma caverna adjacente a ela.

  O agente terá uma regra para cada caverna no seu ambiente

  R1: $\neg f1\text{-}1 \Rightarrow \neg W1\text{-}1 \land \neg W1\text{-}2 \land \neg W2\text{-}1$

  R2: $\neg f2\text{-}1 \Rightarrow \neg W1\text{-}1 \land \neg W2\text{-}1 \land \neg W2\text{-}2 \land \neg W3\text{-}1$

  R3: $\neg f1\text{-}2 \Rightarrow \neg W1\text{-}1 \land \neg W1\text{-}2 \land \neg W2\text{-}2 \land \neg W1\text{-}3$
- O agente também deve saber que, se existe *fedor* em (1,2), então deve haver um Wumpus em (1,2) ou em alguma caverna adjacente a ela:

  R4: $f1\text{-}2 \Rightarrow W1\text{-}3 \lor W1\text{-}2 \lor W2\text{-}2 \lor W1\text{-}1$

Exemplo de Inferência Proposicional no Mundo do Wumpus

- O Wumpus está em (1,3). Como provar isto?
- **O agente precisa mostrar que** W1-3 é consequência lógica da BC. Isto equivale a provar que a sentença BC → W1−3 **é uma sentença** válida:

(1) construindo a Tabela-Verdade para a sentença
- existem 12 símbolos proposicionais na BC, então a Tabela-Verdade terá 12 colunas e 2^{12} = 4096 linhas!
- f1-1, f1-2, f2-1, W1-1, W1-2, W2-1, W2-2, W3-1, W1-3, b1-1, b2-1, b1-2

(2) usando regras de inferência!

Exemplo de Inferência Proposicional no Mundo do Wumpus

¬ f1−1
¬ b1−1
¬ f2−1
b2−1
f1−2
¬ b1−2

W1-3: T ou F **?**

R1: $\neg f1\text{-}1 \Rightarrow \neg W1\text{-}1 \land \neg W1\text{-}2 \land \neg W2\text{-}1$

R2: $\neg f2\text{-}1 \Rightarrow \neg W1\text{-}1 \land \neg W2\text{-}1 \land \neg W2\text{-}2 \land \neg W3\text{-}1$

R3: $\neg f1\text{-}2 \Rightarrow \neg W1\text{-}1 \land \neg W1\text{-}2 \land \neg W2\text{-}2 \land \neg W1\text{-}3$

R4: $f1\text{-}2 \Rightarrow W1\text{-}3 \lor W1\text{-}2 \lor W2\text{-}2 \lor W1\text{-}1$

Exemplo de Inferência Proposicional no Mundo do Wumpus

- Inicialmente, vamos mostrar que o Wumpus não está em nenhuma outra caverna, e então concluir, por eliminação, que ele está em (1,3).

1. Aplicando Modus Ponens a $\neg f1\text{-}1$ e R1, obtemos:
   $\neg W1\text{-}1 \land \neg W1\text{-}2 \land \neg W2\text{-}1$
2. Aplicando E-eliminação a (1), obtemos três sentenças independentes:
   $\neg W1\text{-}1 \quad \neg W1\text{-}2 \quad \neg W2\text{-}1$
3. Aplicando Modus Ponens a $\neg f2\text{-}1$ e R2 , e em seguida aplicando E-eliminação obtemos:
   $\neg W1\text{-}1 \quad \neg W2\text{-}1 \quad \neg W2\text{-}2 \quad \neg W3\text{-}1$
4. Aplicando Modus Ponens a $f1\text{-}2$ e R4, obtemos:
   $W1\text{-}3 \lor W1\text{-}2 \lor W2\text{-}2 \lor W1\text{-}1$

Exemplo de Inferência Proposicional no Mundo do Wumpus

¬ f1−1
¬ b1−1
¬ f2−1
b2−1
f1−2
¬ b1−2
R1: ¬ f1−1 ⇒
¬ W1−1 ∧ ¬ W1−2 ∧ ¬ W2−1
R2: ¬ f2−1 ⇒
¬ W1−1 ∧ ¬ W2−1 ∧ ¬ W2−2 ∧ ¬ W3−1
R3: ¬ f1−2 ⇒
¬ W1−1 ∧ ¬ W1−2 ∧ ¬ W2−2 ∧ ¬ W1−3
R4: f1−2 ⇒
W1−3 ∨ W1−2 ∨ W2−2 ∨ W1−1

5. Aplicando Resolução Unidade, onde β é W1−3 ∨ W1−2 ∨ W2−2 e α é W1−1 obtemos (do passo 2, temos ¬ W1−1):

W1−3 ∨ W1−2 ∨ W2−2

6. Aplicando Resolução Unidade, onde β é W1−3 ∨ W1−2 e α é W2−2 obtemos (do passo 3, temos ¬ W2−2):

W1−3 ∨ W1−2

7. Aplicando Resolução Unidade, onde β é W1−3 e α é W1−2 obtemos (do passo 2, temos ¬ W1−2):

W1−3 !!!

Transformando Conhecimento em Ações

- **O conhecimento inferido deve ser usado para auxiliar o agente a realizar ações.**
- **Deve-se definir regras que relacionem o estado atual do mundo às ações que o agente pode realizar.**
- Ações:
  - avançar para próxima caverna,
  - girar 90 graus à direita ou à esquerda,
  - pegar um objeto na mesma caverna que o agente,
  - atirar na direção para onde o agente está olhando (a flecha para quando encontra uma parede ou mata o Wumpus),
  - sair da caverna.

Transformando Conhecimento em Ações

- Exemplo de Regra:
  - o agente está na caverna (1,1) virado para a direita, e
  - o Wumpus está na caverna (2,1), então:

    A1-1 ∧ Dir ∧ W2-1 ⇒ ¬ avançar
- Com essas regras, o agente pode então perguntar à BC que ação ele deve realizar

Problemas com o Agente Proposicional

- Lógica Proposicional
  - é capaz de fazer inferências que resultam em ações.
  - Contudo, esta lógica é "fraca", não sendo capaz de lidar com domínios simples como o Mundo de Wumpus...
- Problema: existem proposições demais a considerar
  - ex.: a regra: "não avance se o Wumpus estiver em frente a você" só pode ser representada com um conjunto de 64 regras.
  - Assim, serão necessárias milhares de regras para definir um agente eficiente, e o processo de inferência ficará muito lento.

Problemas com o Agente Proposicional

- **Quando o agente faz seu primeiro movimento, a proposição A1-1 torna-se falsa, e A2-1 torna-se verdadeira.**
  - não podemos apenas "apagar" A1-1 porque o agente precisa saber onde esteve antes.
- **Uma solução é usar símbolos diferentes para a localização do agente a cada tempo *t* , contudo...**
  - isso requer regras dependentes do tempo!
  - a BC tem que ser "reescrita" a cada tempo *t*.
- **Se o agente executar 100 passos, a BC terá 6400 regras apenas para dizer que ele não deve avançar quando o Wumpus estiver em frente a ele!**

Uma Solução: Lógica de Primeira Ordem

- Veremos a seguir como construir agentes baseados em Lógica de Primeira Ordem.
- Essa lógica representa objetos e relações entre objetos, além das proposições.
- As 6400 regras do agente proposicional serão reduzidas para 1.

Referências Bibliográficas

- R. Johnsonbaugh. *Discrete Mathematics.* Prentice Hall International, London, UK. 4th. Edition, 1997. Cap. 1.
- M. C. Monard et al. *O cálculo proposicional: uma abordagem voltada à compreensão da linguagem Prolog.* Notas Didáticas do ICMC/USP, (5), Agosto 1992. (http://labic.icmc.sc.usp.br/portugues/SIAE/logica-prolog.html)
- G. Bittencourt. Inteligência Artificial: Ferramentas e Teorias. Editora da UFSC, Florianópolis. 2a. Edição, 2001. Cap. 3.
- S. Russel and P. Norvig. Artificial Intelligence: A Modern Approach. Prentice Hall, Upper Saddle River, USA. 2nd. Edition, 2003. Chapter 7.